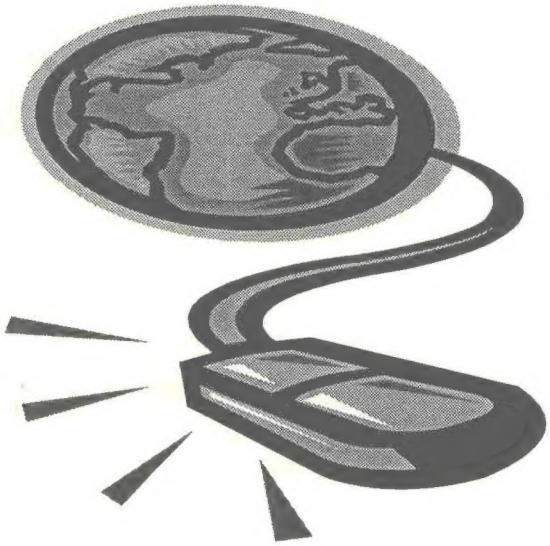
ณรงค์ จันทโพธิ์

SOFT MAIL สร้างสรรค์การใช้งานเม้าส์ด้วย OOP : การสร้าง Line edit



แนวทางในการสร้างออบเจ็กต์ Line edit ที่มีการตอบสนองต่อการกดคีย์บอร์ด

จากที่ผ่านมาทั้ง 4 ตอน ผมได้แนะนำการสร้างแต่ออบเจ็กต์ที่ตอบสนองต่อการกดเม้าส์เป็นหลัก ในตอนนี้เราก็จะมาดูแนวทางในการสร้างออบเจ็กต์ Line edit ที่มีการตอบสนองต่อการกดคีย์บอร์ดซึ่งจะมีความสามารถในการรับข้อมูลจากคีย์บอร์ดทั้งที่เป็นภาษาไทยและภาษาอังกฤษ แบบบรรทัดเดียว

แนวทางการออกแบบ

ในการที่จะเก็บข้อมูลตัวอักษรที่กดคีย์บอร์ดนั้น ออบเจ็กต์ Line edit ก็ต้องได้รับ Focus เสียก่อนที่จะทำการเก็บข้อมูลจากคีย์บอร์ดไว้ ในส่วนนี้จะต้องมีการพิจารณาด้วยว่า ตัวอักษรที่ต้องการจะป้อนนั้นเป็นภาษาไทยหรืออังกฤษ ซึ่งจากตอนที่แล้วผมได้สร้างส่วนของการเปลี่ยนสถานะของคีย์บอร์ดไปแล้ว แต่สิ่งที่ต้องทำในครั้งนี้คือการแปลงข้อมูลของคีย์บอร์ดให้เป็นข้อมูลที่ต้องการว่าจะเป็นการป้อนภาษาไทยหรืออังกฤษ ถ้าพิจารณาที่คีย์บอร์ดแล้วจะพบว่าทั้งตัวอักษรภาษาไทยและภาษาอังกฤษนั้นส่วนมากจะอยู่บนคีย์เดียวกัน ตรงนี้เองครับที่เราจะต้องจัดการกับปัญหาตรงนี้ โดยผมได้สรุปเป็นแนวทางดังนี้ครับ

1. เขียนตัวอักขระบนคีย์บอร์ดทุกตัวที่ใช้ปุ่มเดียวกับภาษาไทย โดยเรียงลำดับตามรหัส ASCII (ต้องไม่ลืมว่ากลุ่มตัวอักษรภาษาอังกฤษนั้นจะมีทั้งตัวพิมพ์ใหญ่และพิมพ์เล็ก รวมทั้งเครื่องหมายสัญลักษณ์พิเศษต่าง ๆ) ซึ่งจะพบว่าค่ารหัส ASCII ที่ได้จะเริ่มต้นจาก 33 -126

2. เขียนตัวอักษรของภาษาไทยโดยจับคู่กับตัวอักษรภาษาอังกฤษ (รวมทั้งสัญลักษณ์พิเศษอื่น ๆ) ที่อยู่บนปุ่มกดเดียวกัน ตัวอย่างเช่น (เรียงตามรหัส ASCII)

!	จะคู่กับ	
5	จะคู่กับ	ถ
A	จะคู่กับ	ฤ
a	จะคู่กับ	ฟ

3. เริ่มเขียนค่ารหัส ASCII ของตัวอักษรทางขวามือ (ใช้ตารางรหัส ASCII ของภาษาไทย สมอ.) โดยทำตามลำดับจากข้อ 2 ซึ่งจะได้ทั้งหมด 94 ค่า ซึ่งค่าตัวเลขทั้ง 94 ค่านี้จะนำมาใช้เป็นตารางแปลงรหัสครับ

สุดท้ายก็จะได้ตารางแปลงรหัสจากภาษาอังกฤษเป็นภาษาไทย ดังลิสติงที่ 1 ครับ

ลิสติงที่ 1

```
unsigned char ThaiChar[94] = {
46,46,242,243,244,219,167,246,247,245,
249,193,162,227,189,168,197,47,45,192,
182,216,214,164,181,171,199,178,170,204,
63,241,196,37,169,175,174,226,172,231,
179,235,201,200,63,236,207,173,240,177,
166,184,234,206,34,41,43,40,186,92,
197,217,248,96,191,212,225,161,211,180,
224,233,195,232,210,202,183,215,185,194,
230,190,203,208,213,205,228,187,209,188,
176,124,39,37};
```

ตารางนี้จะถูกใช้งานเมื่อสถานะการพิมพ์อยู่ในโหมดภาษาไทย เมื่อมีการส่งตัวอักษรให้กับออบเจ็กต์ Line edit จะมีการตรวจสอบค่ารหัส ASCII ของตัวอักษรด้วยคำสั่งดังนี้

```
if(Thai&&ValInRange(Ch,33,126))
   Ch = ThaiChar[Ch-33];
```

นั่นคือเมื่ออยู่ในโหมดภาษาไทยและค่ารหัส ASCII มีค่าอยู่ในช่วง 33-126 จะทำการแปลงรหัสของตัวอักษรนั้นใหม่โดยการคำนวณหาตำแหน่งที่อยู่ของรหัสตัวอักษรใหม่ในตาราง ThaiChar ตัวอย่างเช่น เมื่อเรากด ปุ่มตัว "ก" ซึ่งคอมพิวเตอร์เองจะไม่รู้จักปุ่ม "ก" แต่จะรู้รับเป็นปุ่ม "d" (ไม่ใช่ "D" เพราะไม่มีการกด SHIFT) ซึ่งค่ารหัส ASCII ของตัว "d" นี้ จะมีค่าเป็น 100 ดังนั้นค่าตำแหน่งที่อยู่ของรหัสตัวอักษรใหม่ในตาราง ThaiChar จะมีค่าเท่ากับ 100-33 = 67 ซึ่ง ณ ตำแหน่งที่ 67 นี้ (อย่าลืมว่า ค่าเริ่มต้นของตำแหน่งใน Array ของภาษาซีเริ่มต้นจาก 0) จะมีค่า 161 อยู่ เมื่อเปิดดูตารางรหัส ASCII ที่มีรหัสภาษาไทย (สมอ.) จะพบว่าเป็น "ก" นั่นคือขณะนี้ Line edit จะเก็บตัวอักษรใหม่ในรหัสของ "ก"

ลิสติงที่ 2 LNEDIT.CPP

```
#include"graphics.h"
#include"mouobj.h"
#include"stdfon.h"
#include"objects.h"
extern WORD MouEvent;
extern int MouX,MouY;
extern int Thai;
extern unsigned char ThaiChar[];
//*************************************************************************
LineEdit::LineEdit(int X,int Y,char *title,int color,int Wide,int MaxLen,
            int style,unsigned int RefId){
            ST_LEN      = MaxLen;
            ObjType     = LINE_EDIT;
            Title       = new char[strlen(title)+1];
            strcpy(Title,title);
            Text        = new char[ST_LEN];
            Text[0]     = '\0';
            SCol        = X;
            SRow        = Y;
            EditPos     = ThaiLen(Title)*8+8;
            ECol        = SCol+Wide*8+8 + EditPos;
            ERow        = SRow+22;
            CurCol      = SCol+4+EditPos;
            Style       = style;
            Color       = color;      //Text Color
            TextBk      = WHITE;
            Index       = 0;          //Use for Arrow key,Del key was pressed
            CurNum      = 0;          //Character counter  (0 - ST_LEN)
            RetCmd      = RefId;
            Cmd         = 0x0000;
            if(Style&VISIBLE)
            Draw();
            if(Style&GRAYED)
            Style^      =GRAYED;
}
 //*************************************************************************
 void LineEdit::Draw(){
    Style|=VISIBLE;
    HideMouse();
    setfillstyle(1,getpixel(SCol-1,SRow+4));
    bar(SCol,SRow,SCol+EditPos,ERow);
    OutThaiXY(SCol,SRow+5,Title,Color);
    setcolor(BLACK);
    rectangle(SCol+EditPos,SRow,ECol,ERow);
    setfillstyle(1,TextBk);
    bar(SCol+EditPos+1,SRow+1,ECol-1,ERow-1);
    OutThaiXY(SCol+12+EditPos,SRow+5,Text,BLACK);
    ShowMouse();
    if(Style&FOCUSED)
       DrawCur();
}
 //*************************************************************************
 char *LineEdit::SendStr(){
    return Text;
}
```

→

ลิสติงที่ 2 (ต่อ)

```
56:    //*************************************************************************
57:    void LineEdit::GetStr(BYTE *Str){
58:    int ULCount       =0;
59:       CurNum         = strlen(Str);
60:       strcpy(Text,Str);
61:       for(Index      =0;Index<CurNum;Index++)
62:          if(UpLoSymbols(Text[Index]))
63:    ULCount++;
64:       CurCol         = SCol + 4+EditPos + (CurNum-ULCount)*8;
65:       Index          = CurNum;
66:       Draw();
67:    }
68:    //*************************************************************************
69:    void LineEdit::Destroyed(){
70:       delete Text;
71:       delete Title;
72:    }
73:    //*************************************************************************
74:    int LineEdit::WaitRelease(){
75:       while((MouEvent==L_DOWN)&&MouInBox(MouX,MouY,SCol,SRow,ECol,ERow))
76:          GetMouInfo(&MouEvent,&MouX,&MouY);
77:       if(MouInBox(MouX,MouY,SCol,SRow,ECol,ERow))
78:          return 1;
79:       return 0;
80:    }
81:    //*************************************************************************
82:    int LineEdit::Pressed(){
83:       if(MouEvent==L_DOWN&&MouInBox(MouX,MouY,SCol,SRow,ECol,ERow)&&WaitRelease()){
84:          if(!(Style&FOCUSED))
85:    DrawCur();
86:          return 1;
87:          }
88:       else
89:          return 0;
90:    }
91:    //*************************************************************************
92:    void LineEdit::DrawButtUp(){
93:       EraseCur();
94:    }
95:    //*************************************************************************
96:    void LineEdit::OutChar(char Ch){//Display Character at Current collumn
97:       HideMouse();
98:       if(Style&PASSWRD)
99:          PutChar(CurCol,SRow+5,'*',BLACK);
100:       else
101:       if(IsHiAboveSymbol(Ch)&&IsAboveSymbol(Text[Index-2]))
102:          PutChar(CurCol-8,SRow+1,Ch,BLACK);
103:       else
104:       if(IsAboveSymbol(Ch))
105:          PutChar(CurCol-8,SRow+5,Ch,BLACK);
106:       else
107:       if(IsBelowSymbol(Ch))
108:          PutChar(CurCol-8,SRow+6,Ch,BLACK);
109:       else
110:          PutChar(CurCol,SRow+5,Ch,BLACK);
111:       ShowMouse();
112:    }
113:    //*************************************************************************
114:    void LineEdit::EraseChar(){//Erase at Current collumn (CurCol)
115:       HideMouse();
116:       setfillstyle(1,TextBk);
117:       if(Style&PASSWRD)
118:          bar(CurCol,SRow+1,CurCol+8,ERow-1);
119:       else
120:       if(IsHiAboveSymbol(Text[Index])){
121:          if(IsAboveSymbol(Text[Index-1]))
122:    bar(CurCol,SRow+1,CurCol+8,SRow+4);
123:          else
124:    bar(CurCol,SRow+1,CurCol+8,SRow+7);
125:          }
126:       else
127:       if(IsAboveSymbol(Text[Index])){
```

→

ลิสติงที่ 2 (ต่อ)

```
128:          bar(CurCol,SRow+1,CurCol+8,SRow+7);
129:          }
130:       else
131:       if(IsBelowSymbol(Text[Index]))
132:          bar(CurCol,ERow-3,CurCol+8,ERow-1);
133:       else
134:          bar(CurCol,SRow+1,CurCol+8,ERow-1); //Erase previous character
135:       ShowMouse();
136:    }
137:    //*************************************************************************
138:    void LineEdit::DrawCur(){ //Show cursor current collumn
139:       HideMouse();
140:       setcolor(BLACK);
141:       line(CurCol,SRow+17,CurCol+8,SRow+17);
142:       ShowMouse();
143:    }
144:    //*************************************************************************
145:    void LineEdit::EraseCur(){//Erase cursor at current collumn (CurCol)
146:       HideMouse();
147:       setcolor(TextBk);
148:       line(CurCol,SRow+17,CurCol+8,SRow+17);
149:       ShowMouse();
150:    }
151:    //*************************************************************************
152:    void LineEdit::ShowCurXY(){ //Show cursor at current collumn
153:       EraseCur();
154:       CurCol+=8;
155:       DrawCur();
156:    }
157:    //*************************************************************************
158:    void LineEdit::ArrLeft(){
159:    }
160:    //*************************************************************************
161:    void LineEdit::ArrRight(){
162:    }
163:    //*************************************************************************
164:    void LineEdit::Delete(){  //Del key was pressed
165:    }
166:    //*************************************************************************
167:    void LineEdit::MoveStr(){ //For Insert mode and Del key was pressed
168:    }
169:    //*************************************************************************
170:    void LineEdit::BackSpace(){//delete character at previous CurCol
171:       if((CurNum>0)&&(Index>0)){
172:          EraseCur();
173:          CurNum--;
174:          Index--;
175:          CurCol-=8;
176:          EraseChar();
177:          if(UpLoSymbols(Text[Index])&&(!(Style&PASSWRD)))
178:    CurCol+=8;
179:          DrawCur();
180:          Text[CurNum] = '\0';
181:          }
182:       else
183:          putch(7);
184:    }
185:    //*************************************************************************
186:    void LineEdit::GetKey(WORD InKey){
187:    unsigned char Ch=InKey&0x00FF;      //read low byte of the pressed key
188:      switch(Ch) {
189:      case 0 : Ch = InKey>>8; //read high byte of the pressed key
190:            switch(Ch) {
191:               case  75  :  ArrLeft();
192:                            break;
193:               case  77  :  ArrRight();
194:                            break;
195:               }
196:            break;
197:          case  0x08 :  BackSpace();
198:            break;
199:          default    :  if(Index < (ST_LEN-1)&&(Ch>31)){
200:               if(Style&NUMER){
```

→

```
201:                if(ValInRange(Ch,48,57)&&(!Thai));
202:                else{
203:                   putch(7);
204:                   break;
205:                   }
206:                }
207:               if(Thai&&ValInRange(Ch,33,126))
208:                Ch = ThaiChar[Ch-33];
209:               Text[Index] = Ch;
210:               if(CurNum)
211:                EraseCur();
212:               Index++;
213:               CurNum++;
214:               Text[CurNum] = '\0';
215:               OutChar(Ch);
216:               if((!UpLoSymbols(Ch))||(Style&PASSWRD))
217:                CurCol+=8;
218:                DrawCur();
219:         }
220:                else
221:              putch(7);
222:          }
223:     }
224:     //*******************************************
```

ลิสติงที่ 2 (ต่อ)

เมื่อเข้าใจในส่วนการแปรรหัสภาษาแล้ว ก็มาเริ่มดูรายละเอียดการสร้างออบเจ็กต์ Line edit กันเลยครับ รายละเอียดอยู่ในลิสติงที่ 2 ในการสร้างตัวออบเจกต์จะใช้ Constructor LineEdit() เป็นตัวกำหนดรายละเอียดเริ่มต้น ดังรายละเอียดของ Parameter ดังนี้ครับ

- Col,Row เป็นตำแหน่ง x,y ที่จะทำการวาดออบเจ็กต์
- title เป็นชื่อของ Line edit
- color เป็นสีของชื่อของ Line edit
- Wide เป็นจำนวนของตัวอักษรที่สูงสุดที่จะแสดงให้เห็นได้
- MaxLen เป็นจำนวนตัวอักษรทั้งหมดที่ Line edit จะเก็บไว้ได้
- style เป็นค่าสถานะการตอบรับการทำงานของออบเจ็กต์
- RefId เป็นค่าอ้างอิงที่จะใช้กับออบเจ็กต์

สำหรับ style ที่กำหนดให้ Line edit นั้น จะมี 3 แบบ

NUMER เป็น Line edit ที่จะคอยรับข้อมูลที่เป็นตัวเลขเท่านั้น ถ้าหากมีการกดตัวอักษรอื่นที่ไม่ใช่ตัวเลขแล้วจะไม่ยอมรับพร้อมกับมีเสียงเตือน

PASSWRD เป็น Line edit ที่จะไม่แสดงตัวอักษรที่ป้อนเข้าไป แต่จะแสดงเพียงตัว "*" ออกมาเพียงอย่างเดียว สำหรับ style แบบนี้จะใช้ในการรับข้อมูลที่เป็นความลับเช่น รหัสผ่าน (Password) เป็นต้น

แบบธรรมดา ออบเจ็กต์แบบนี้เป็น Line edit ที่ประกาศ style แบบ VISIBLE เท่านั้น ซึ่งจะใช้รับข้อมูลทั่ว ๆ ไปได้โดยไม่จำกัดชนิด

สำหรับในการส่งและการอ่านข้อมูลจาก Line edit นั้นจะอาศัย Method ที่ประกาศเพิ่มใน class GUIApp ซึ่งมีรายละเอียดดังนี้ครับ

```
void GUIApp::SendIntToLnEdit(int Val,int RefId) ;
```

เป็น Method ที่ทำหน้าที่ในการส่งค่าที่เป็นตัวเลขให้กับ Line edit โดยมีรายละเอียดของพารามิเตอร์ดังนี้

- Val เป็นค่าตัวเลขที่ต้องการส่ง
- RefId เป็นค่าอ้างอิงของ Line edit

```
void GUIApp::SendStrToLnEdit(BYTE*Str,int RefId)
```

เป็น Method ที่ทำหน้าที่ในการส่งข้อความให้กับ Line edit โดยมีรายละเอียดของพารามิเตอร์ดังนี้

- Str เป็นข้อความที่ต้องการส่ง
- RefId เป็นค่าอ้างอิงของ Line edit

```
int GUIApp::GetLnEditInt(int RefId)
```

เป็น Method ที่ทำหน้าที่ในการอ่านข้อมูลตัวเลขจาก Line edit โดยมีรายละเอียดของพารามิเตอร์ดังนี้

- RefId เป็นค่าอ้างอิงของ Line edit

```
BYTE *GUIApp::GetLnEditStr(int RefId)
```

เป็น Method ที่ทำหน้าที่ในการอ่านข้อความจาก Line edit โดยมีรายละเอียดของพารามิเตอร์ดังนี้

- RefId เป็นค่าอ้างอิงของ Line edit

ข้อควรระวังในการใช้งานทั้ง 4 ฟังก์ชันนี้อยู่ตรงที่ ควรจะใช้ให้ตรงกับ style ของ Line edit คือ Line edit ที่ มี style เป็น NUMER ก็ต้องใช้ฟังก์ชัน SendIntToLnEdit() และ GetLnEditInt() เท่านั้น

ปรับปรุงโค้ดเก่า

ในการที่จะทำให้ Line edit สามารถรับข้อมูลตัวอักษรที่เกิดจากการกดคีย์บอร์ดได้นั้นจะต้องมีการจัดการกับส่วนของการดัก Message ที่เกิดจากการกดคีย์บอร์ดเสียก่อน นั่นคือทุกครั้งที่มีการกดคีย์บอร์ดจะต้องทำการตรวจสอบดูว่า ในขณะนั้น ออบเจ็กต์ที่ได้โฟกัสนั้นเป็น Line edit หรือไม่ ถ้าใช่ต้องทำการส่งข้อมูลตัวอักษรที่เกิดจากการกด

คีย์บอร์ดไปให้กับ Line edit นั้น รายละเอียดของการทำงานอยู่ในส่วนท้ายของหัวข้อการแก้ไข ไฟล์ OBJINTER.CPP

OBJINTER.H

ใน class GUIApp เพิ่มการประกาศ Method

```
BYTE *GetLnEditStr(int );
int  GetLnEditInt(int);
void SendStrToLnEdit(BYTE *,int );
void SendIntToLnEdit(int ,int);
```

OBJINTER.CPP

เพิ่มฟังก์ชันการทำงานจากลิสติงที่ 3

และใน Method GetMessage() ที่ส่วนล่างของการตรวจสอบ Message ที่เกิดจากการกดคีย์บอร์ดจาก...

```
if(CurrObj==NULL) //no window
   Thai^=1;       //set system language
return ID_UNDEF;
}
while(CurrObj!=NULL){
```

ให้เพิ่มส่วนการทำงานตรวจสอบการได้รับ Focus ของ Line edit ดังนี้

```
if((CurrObj->ObjType==LINE_EDIT)&&(CurrObj->Style&FOCUSED)){
CurrObj->GetKey(Msg);
}
else
```

และส่วนท้ายสุดของการตรวจสอบ Message ของคีย์บอร์ดจาก...

```
return Msg;
}
else {   //Mouse was pressed
while((CurrObj!=NULL)&&(MouEvent!=NO_DOWN)){
```

ให้เปลี่ยนจาก return Msg; เป็น return ID_UNDEF; แทน ซึ่งในส่วนนี้จะเป็นการป้องกันการส่งข้อมูลตัวอักษร ที่เกิดจากการป้อนให้กับ Line edit ออกไปสู่ Message Loop

STDFON.H

เพิ่มการประกาศฟังก์ชันอีกดังนี้ครับ

```
int UpLoSymbols(unsigned char);
```

ลิสติงที่ 3

```
1:   void GUIApp::SendIntToLnEdit(int Val,int RefId){
2:   BYTE *Str = new char[5];
3:   AppObj *CurrObj = FirstObj;
4:      itoa(Val,Str,10);
5:      while(CurrObj!=NULL){
6:         if((CurrObj->ObjType==LINE_EDIT)&&(CurrObj->RetCmd==RefId)){
7:     CurrObj->GetStr(Str);
8:     return;
9:     }
10:         CurrObj=CurrObj->NextObj;
11:         }
12:      delete Str;
13:  }
14:  //*****************************************************************************
15:  void GUIApp::SendStrToLnEdit(BYTE *Str,int RefId){
16:  AppObj *CurrObj = FirstObj;
17:     while(CurrObj!=NULL){
18:        if((CurrObj->ObjType==LINE_EDIT)&&(CurrObj->RetCmd==RefId)){
19:    CurrObj->GetStr(Str);
20:    return;
21:    }
22:         CurrObj=CurrObj->NextObj;
23:         }
24:  }
25:  //*****************************************************************************
26:  int  GUIApp::GetLnEditInt(int RefId){
27:     return atoi(GetLnEditStr(RefId));
28:  }
29:  //*****************************************************************************
30:  BYTE *GUIApp::GetLnEditStr(int RefId){
31:  AppObj *CurrObj = FirstObj;
32:     while(CurrObj!=NULL){
33:        if((CurrObj->ObjType==LINE_EDIT)&&(CurrObj->RetCmd==RefId))
34:    return CurrObj->SendStr();
35:         CurrObj=CurrObj->NextObj;
36:         }
37:     return NULL;
38:  }
```

เป็นฟังก์ชันที่ทำหน้าที่ตรวจสอบตัวอักษรที่ส่งมาเป็นสระบน-ล่าง หรือวรรณยุกต์หรือไม่ ถ้าใช่จะ return 1 ออกมา

```
int IsHiAboveSymbol(unsigned char);
```

เป็นฟังก์ชันที่ทำหน้าที่ตรวจสอบตัวอักษรที่ส่งมาเป็นวรรณยุกต์หรือไม่ ถ้าใช่จะ return 1 ออกมา สำหรับวรรณยุกต์ในกรณีนี้ จะอยู่ในลักษณะที่ตามหลังสระบนได้ ได้แก่ ไม้เอก โท ตรี จัตวา รวมทั้ง การันต์ ด้วย

```
int IsAboveSymbol(unsigned char);
```

เป็นฟังก์ชันที่ทำหน้าที่ตรวจสอบตัวอักษรที่ส่งมาเป็น สระบน หรือไม่ ถ้าใช่จะ return 1 ออกมา

```
int IsBelowSymbol(unsigned char);
```

เป็นฟังก์ชันที่ทำหน้าที่ตรวจสอบตัวอักษรที่ส่งมาเป็น สระล่าง ได้แก่ สระอุและสระอู หรือไม่ ถ้าใช่จะ return 1 ออกมา

ฟังก์ชันทั้ง 4 นี้จะใช้ช่วยในการทำงานในส่วนการจัดการกับตำแหน่งที่ถูกต้องของสระและวรรณยุกต์ดังกล่าว

```
void PutChar(int X,int Y,unsigned char Asc,int Color);
```

เป็นฟังก์ชันที่ทำหน้าที่แสดงตัวอักษรทีละตัวเช่นเดียวกับ DisplayChar() เพียงแต่สามารถกำหนดสีของตัวอักษรได้อีกด้วย

STDFON.CPP

ให้เพิ่มการประกาศตารางภาษาไทยจากลิสติงที่ 1
เพิ่มส่วนการทำงานของฟังก์ชันดังลิสติงที่ 4

OBJECTS.H

เพิ่มการประกาศ ตัวแปรดังนี้

```
#define     LINE_EDIT   5
const WORD NUMER        = 32;
const WORD PASSWRD      = 64;
```

ใน class AppObj เพิ่ม Method... ดังต่อไปนี้เพื่อทำให้ Base class รู้จัก Method ภายในของ ออบเจ็กต์ Line edit

```
virtual void GetKey(WORD){};
virtual void GetStr(BYTE *){};
virtual char *SendStr(){return 0;};
```

เพิ่มการประกาศออบเจ็กต์ Line edit ดังในลิสติงที่ 5

ลิสติงที่ 4

```
1:  int UpLoSymbols(unsigned char Symbol){
2:     if(ValInRange(Symbol,212,217)||ValInRange(Symbol,231,236)||(Symbol==209))
3:        return 1;
4:     return 0;
5:  }
6:  /*****************************************************************************/
7:  int IsHiAboveSymbol(unsigned char Symbol){
8:     if(ValInRange(Symbol,232,236))
9:        return 1;
10:    return 0;
11: }
12: /*****************************************************************************/
13: int IsAboveSymbol(unsigned char Symbol){
14:    if(ValInRange(Symbol,212,215)||(Symbol==209)||IsHiAboveSymbol(Symbol)||
15:      (Symbol==231))
16:      return 1;
17:    return 0;
18: }
19: /*****************************************************************************/
20: int IsBelowSymbol(unsigned char Symbol){
21:    if(ValInRange(Symbol,216,217))
22:       return 1;
23:    return 0;
24: }
25: /*****************************************************************************/
26: void PutChar(int X,int Y,unsigned char Asc,int Color){
27: int i,j;
28:    for(i=0;i<16;i++){
29:       for(j=0;j<8;j++){
30:  if(StdFont[Asc][i]&(128>>j))
31:     putpixel(X+j,Y,Color);
32:  }
33:       Y++;
34:       }
35: }
```

ลิสติงที่ 5

```
1:  class LineEdit : public AppObj {
2:  public :
3:  int TextBk,CurCol,Index,CurNum;
4:  int ST_LEN;
5:  char *Text;
6:  int EditPos;
7:  LineEdit(int,int,char *,int,int,int,int,unsigned int);
8:  void Destroyed();
9:  void OutChar(char);
10: void EraseChar();
11: void EraseCur();
12: void DrawCur();
13: void Draw();
14: void ShowCurXY();
15: void ArrLeft();
16: void ArrRight();
17: void MoveStr();
18: void BackSpace();
19: void Delete();
20: void GetKey(WORD);
21: void GetStr(BYTE *);
22: char *SendStr();
23: int Pressed();
24: int WaitRelease();
25: void DrawButtUp();
26: }; //Class LineEdit
```

ทดลองใช้งาน

โปรแกรมตัวอย่างที่ใช้ในการทดลองใช้งานออบเจ็กต์ Line edit นี้รายละเอียดดังในลิสติงที่ 6

รายละเอียดการทำงานของโปรแกรมมีดังนี้ครับ

บรรทัดที่ 8-18 เป็นการประกาศ Message ที่จะให้มีการทำงาน

บรรทัดที่ 25-37 เป็นฟังก์ชันที่ใช้แสดงผลจากการอ่านค่าในออบเจ็กต์ Line edit ของโปรแกรม

บรรทัดที่ 39-60 เป็นฟังก์ชันที่ทำหน้าที่หาข้อความประจำแต่ละ Message ที่ส่งเข้ามาโดยที่บรรทัดที่ 44-48

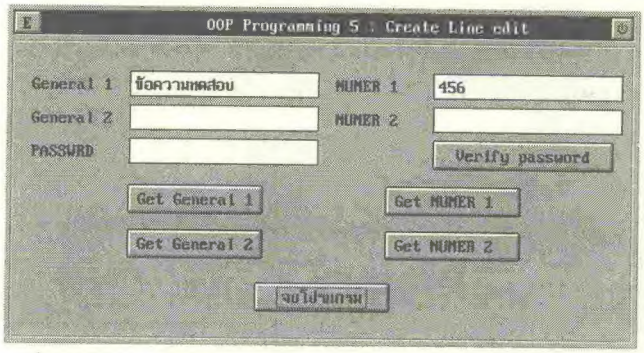
เป็นการตรวจสอบค่ารหัสผ่านจาก ออบเจ็กต์ Line edit ที่มี style แบบ PASSWRD ในโปรแกรมผมได้กำหนดรหัสเป็น 1234 ส่วนบรรทัดที่ 51-58 จะอ่านข้อความและค่าที่มีอยู่ใน Line edit แบบธรรมดาและแบบ NUMER

บรรทัดที่ 62-72 เป็นฟังก์ชันที่นำเอา Message ที่ได้รับมาตรวจสอบว่าเป็น Message ที่สนใจหรือไม่ เพื่อที่จะไปทำงานตามที่กำหนดต่อไป

บรรทัดที่ 80-105 เป็นส่วนของ main ฟังก์ชันที่ทำการสร้างออบเจ็กต์ต่าง ๆ ที่จะใช้ในโปรแกรม

บรรทัดที่ 110-114 เป็นส่วนของ Message Loop

สำหรับการคอมพายล์ต้องเปิดเป็นโปรเจ็กต์เช่นเดิมครับ สำหรับในครั้งนี้ในโปรเจ็กต์จะประกอบด้วยไฟล์ต่อไปนี้ครับ



รูปที่ 1 แสดงหน้าจอการทำงานของโปรแกรมตัวอย่าง

1.SAMPLE.CPP
2.STDFON.CPP
3.MOUOBJ.CPP
4.OBJINTER.CPP
5.BUTTON.CPP
6.CHECKBOX.CPP
7.RADIOBTN.CPP
8.WINDOW.CPP
9.LNEDIT.CPP

หากท่านผู้อ่านท่านใดไม่มีโค้ดจากตอนเก่า ก็คงต้องติดต่อใช้บริการซอฟต์เมล์ได้ครับ สำหรับในแผ่นซอฟต์เมล์ จะประกอบไปด้วยไฟล์ต่าง ๆ ดังนี้ครับ

1.OBJECTS.H
2.OBJINTER.H
3.MOUOBJ.H
4.STDFON.H
5.THAIENG1.TXT
6.STDFON.CPP
7.MOUOBJ.CPP
8.RADIOBTN.CPP
9.CHECKBOX.CPP
10.BUTTON.CPP
11.WINDOW.CPP
12.LNEDIT.CPP
13.OBJINTER.CPP
14.SAMPLE.CPP

โดยที่รายการ 1, 2, 4, 6, 13 เป็นไฟล์เก่าที่ได้ปรับปรุงตามเนื้อหาในตอนนี้

รายการ 12, 14 เป็นไฟล์ใหม่จากเนื้อหาในตอนนี้

รายการ 3, 5, 7, 8, 9, 10, 11 เป็นไฟล์เก่าจากตอนที่ 4

ลิสติงที่ 6 SAMPLE.CPP

```
SAMPLE.CPP
#include<graphics.h>
#include<string.h>
#include<stdlib.h>
#include"mouobj.h"
#include"stdfon.h"
#include"objects.h"
#include"objinter.h"
#define ID_QUIT          10
#define ID_TBUTT1        101
#define ID_TBUTT2        102
#define ID_TBUTT3        103
#define ID_TBUTT4        104
#define ID_TBUTT5        105
#define LNEDIT_NORMAL1   110
#define LNEDIT_NORMAL2   111
#define LNEDIT_NUMER1    112
#define LNEDIT_NUMER2    113
#define LNEDIT_PASSWRD   114
 unsigned int MouEvent;
 int     MouX,MouY;
 GUIApp TestApp;
 WORD    AppMsg;
 int     Quit = 0;
 //*************************************************************************
 void ShowMessage(BYTE *MsgStr){
 GUIApp ShowMsg;
 WORD TestMsg;
    ShowMsg.AddObj(new Window("Show message",130,80,350,200,VISIBLE|SYS_CLOSE|
                   CAPTION,WHITE,BLUE,LIGHTGRAY));
    ShowMsg.AddObj(new TextButton(282,235,60,25,"Ok",VISIBLE|FOCUSED,27,CLOSEWIN));
    OutThaiXY(155,170,MsgStr,YELLOW);
    ShowMouse();
    while(TestMsg!=CLOSEWIN){          //Wait for CLOSEWIN Message
       GetKbdMouse(&MouEvent,&MouX,&MouY);
       TestMsg = ShowMsg.GetMessage(MouEvent,MouX,MouY);
       }
 }
 //*************************************************************************
      BYTE *GetButtMsg(WORD Msg){
      BYTE *Str = new char[150];
      BYTE *ButtMsg[] = {"General 1 ","General 2 ","NUMER 1 ","NUMER 2 "};
      BYTE Tstr[10];
     if(Msg==ID_TBUTT5){    //Verify password
        if(strcmp("1234",TestApp.GetLnEditStr(LNEDIT_PASSWRD))==0)
      return "              Password ok";
        else
      return "           Password invalid";
        }
     strcpy(Str,ButtMsg[Msg-101]);
     strcat(Str,": ");
     if((Msg==ID_TBUTT1)||(Msg==ID_TBUTT2))
        strcat(Str,TestApp.GetLnEditStr(Msg+9));  //Msg+9 = LNEDIT_NORMAL1
                                  //or LNEDIT_NORMAL2
     else{
        itoa(TestApp.GetLnEditInt(Msg+9),Tstr,10);//LNEDIT_NUMER1
                                  //or LNEDIT_NUMER2
        strcat(Str,Tstr);
        }
     return Str;
 }
 //*************************************************************************
 void DoMessage(WORD *Msg){
    switch(*Msg){
       case ID_QUIT : case CLOSEWIN :
            Quit = 1;
            break;
       case ID_TBUTT1 : case ID_TBUTT2 : case ID_TBUTT3 :
       case ID_TBUTT4 : case ID_TBUTT5 :
            ShowMessage(GetButtMsg(*Msg));
            break;
       }
 }
```

ลิสติงที่ 6 (ต่อ)

```
//*****************************************************************************
main(){
   int Gd=DETECT,Gm;
   initgraph(&Gd,&Gm,"c:\\lang\\bc31\\bgi");
   InitMouse();
   setrgbpalette(LIGHTGRAY,48,48,48);
   setrgbpalette(DARKGRAY+48,35,35,35);
   TestApp.AddObj(new Window("OOP Programming 5 : Create Line edit",0,0,639,479,
                  VISIBLE|LANG_KEY|SYS_CLOSE,WHITE,BLUE,LIGHTGRAY));
   //create text button objects
     TestApp.AddObj(new TextButton(150,270,120,25,"Get General 1",VISIBLE,0x00,
                    ID_TBUTT1));
     TestApp.AddObj(new TextButton(150,310,120,25,"Get General 2",VISIBLE,0x00,
                    ID_TBUTT2));
     TestApp.AddObj(new TextButton(400,270,120,25,"Get NUMER 1  ",VISIBLE,0x00,
                    ID_TBUTT3));
     TestApp.AddObj(new TextButton(400,310,120,25,"Get NUMER 2  ",VISIBLE,0x00,
                    ID_TBUTT4));
     TestApp.AddObj(new TextButton(275,370,120,25,"¯ªâ»Ãá¡ÃÁ",VISIBLE|FOCUSED,
                    0x2d00,ID_QUIT));
     TestApp.AddObj(new TextButton(400,160,160,25,"Verify password",VISIBLE,
                    0x00,ID_TBUTT5));
   //create line edit objects
     TestApp.AddObj(new LineEdit(30,100, "General 1",BLUE,20,20,VISIBLE,
                    LNEDIT_NORMAL1));
     TestApp.AddObj(new LineEdit(30,130,"General 2",BLUE,20,20,VISIBLE,
                    LNEDIT_NORMAL2));
     TestApp.AddObj(new LineEdit(320,100, "NUMER 1  ",BLUE,20,20,VISIBLE|NUMER,
                    LNEDIT_NUMER1));
     TestApp.AddObj(new LineEdit(320,130,'NUMER 2  ",BLUE,20,20,VISIBLE|NUMER,
                    LNEDIT_NUMER2));
     TestApp.AddObj(new LineEdit(30,160,"PASSWRD  ",BLUE,20,20,VISIBLE|PASSWRD,
                    LNEDIT_PASSWRD));
    //initial line edit 's value
     TestApp.SendStrToLnEdit("ข้อความทดสอบ",LNEDIT_NORMAL1);
     TestApp.SendIntToLnEdit(456,LNEDIT_NUMER1);
    ShowMouse();
    while(!Quit){
       GetKbdMouse(&MouEvent,&MouX,&MouY);
       AppMsg = TestApp.GetMessage(MouEvent,MouX,MouY);
       DoMessage(&AppMsg);
       }
    HideMouse();
    closegraph();
    return 0;
  }
```

■

ส่งท้าย

จากลิสติงที่ 2 จะพบว่ามีหลายเมธอดที่ว่างเปล่าอยู่ ได้แก่

```
void ArrLeft();
void ArrRight();
void MoveStr();
void Delete();
```

ในส่วนนี้เป็นส่วนที่ผมเขียนขึ้นมารองรับในการใช้งานปุ่มลูกศร, Del, Insert ช่วยในการทำงานของ Line edit ซึ่งจะช่วยให้มีการ scroll ข้อความในออบเจ็กต์ได้แต่ยังไม่มีคำสั่งการทำงานใด ๆ อย่างไร ผมขอฝากให้ท่านผู้อ่านไปลองเขียนกันดูก็แล้วกันครับ และเมื่อยังไม่มีการ scroll ได้ ดังนั้นในการประกาศสร้างออบเจ็กต์ Line edit ในส่วนของ Wide และ MaxLen นั้นควรจะมีค่าเท่ากัน

และสำหรับท่านผู้อ่านท่านใดมีปัญหาหรือคำแนะนำประการใดเกี่ยวกับบทความ สามารถติดต่อผมได้ที่

Email:u3500878@ram1.ru.ac.th (http://www.ru.ac.th/~u3500878/)

สวัสดีครับ